Contents

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This script creates the imu objects, the gps object, and the
% insFilterMARG object for the three agents. Agent 1 has is
% equipped with both an IMU and GPS while Agents 2 & 3 have
% only an IMU (with magnetometer). Currently all 3 IMU have
% the same properties with different RNG seeds but this may
% be adjusted if desired. Agent 1 uses MATLAB's insfilterMARG
% for optimal state estimation and sensor fusion. Agents 2/3
% will use our custom EKF that utilizes GPS measurements from
% Agent 1 to improve their state estimates
%
% Generates one of two reference trajectories in the NED frame
% for the three Agents. Moving traj from LoggedQuadcopter.mat
% with Agents 2/3 offset positionally 5 m North and East
% respectively. Stationary traj starts at the same initial
% position as Moving traj but has zero vel or orientation change
%
% Author: Nathan Gurgens
% Credit to MATLAB Example:
% https://www.mathworks.com/help/fusion/ug/imu-and-gps-fusion-for-inertial-navigation.html
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

IMU_1 setup

```
fs_imu = 160; % Hz

imu_1 = imuSensor('accel-gyro-mag', 'SampleRate', fs_imu);
imu_1.MagneticField = [19.5281 -5.0741 48.0067];

% Accelerometer
imu_1.Accelerometer.MeasurementRange =  19.6133;
imu_1.Accelerometer.Resolution = 0.0023928;
imu_1.Accelerometer.ConstantBias = 0.19;
imu_1.Accelerometer.NoiseDensity = 0.0012356;

% Gyroscope
imu_1.Gyroscope.MeasurementRange = deg2rad(250);
imu_1.Gyroscope.Resolution = deg2rad(0.0625);
```

```
imu_1.Gyroscope.ConstantBias = deg2rad(3.125);
imu_1.Gyroscope.AxesMisalignment = 1.5;
imu_1.Gyroscope.NoiseDensity = deg2rad(0.025);

% Magnetometer
imu_1.Magnetometer.MeasurementRange = 1000;
imu_1.Magnetometer.Resolution = 0.1;
imu_1.Magnetometer.ConstantBias = 100;
imu_1.Magnetometer.NoiseDensity = 0.3/ sqrt(50);

% Random Number Generator
imu_1.RandomStream = 'mt19937ar with seed';
imu_1.Seed = 1;
```

IMU_2 setup

```
imu_2 = imuSensor('accel-gyro-mag', 'SampleRate', fs_imu);
imu_2.MagneticField = [19.5281 -5.0741 48.0067];

% Accelerometer
imu_2.Accelerometer.MeasurementRange =  0.99*19.6133;
imu_2.Accelerometer.Resolution = 0.99*0.0023928;
imu_2.Accelerometer.ConstantBias = 0.99*0.19;
imu_2.Accelerometer.NoiseDensity = 0.99*0.0012356;

% Gyroscope
imu_2.Gyroscope.MeasurementRange = 0.99*deg2rad(250);
imu_2.Gyroscope.Resolution = 0.99*deg2rad(0.0625);
imu_2.Gyroscope.ConstantBias = 0.99*deg2rad(3.125);
imu_2.Gyroscope.AxesMisalignment = 0.99*1.5;
imu_2.Gyroscope.NoiseDensity = 0.99*deg2rad(0.025);

% Magnetometer
imu_2.Magnetometer.MeasurementRange = 0.99*1000;
imu_2.Magnetometer.Resolution = 0.99*0.1;
imu_2.Magnetometer.ConstantBias = 0.99*100;
imu_2.Magnetometer.NoiseDensity = 0.99*0.3/ sqrt(50);

% Random Number Generator
imu_2.RandomStream = 'mt19937ar with seed';
imu_2.Seed = 2;
```

IMU_3 setup

```
imu_3 = imuSensor('accel-gyro-mag', 'SampleRate', fs_imu);
imu_3.MagneticField = [19.5281 -5.0741 48.0067];

% Accelerometer
imu_3.Accelerometer.MeasurementRange =  1.01*19.6133;
imu_3.Accelerometer.Resolution = 1.01*0.0023928;
imu_3.Accelerometer.ConstantBias = 1.01*0.19;
imu_3.Accelerometer.NoiseDensity = 1.01*0.0012356;

% Gyroscope
imu_3.Gyroscope.MeasurementRange = 1.01*deg2rad(250);
imu_3.Gyroscope.Resolution = 1.01*deg2rad(0.0625);
imu_3.Gyroscope.ConstantBias = 1.01*deg2rad(3.125);
imu_3.Gyroscope.AxesMisalignment = 1.01*1.5;
```

```
imu_3.Gyroscope.NoiseDensity = 1.01*deg2rad(0.025);

% Magnetometer
imu_3.Magnetometer.MeasurementRange = 1.01*1000;
imu_3.Magnetometer.Resolution = 1.01*0.1;
imu_3.Magnetometer.ConstantBias = 1.01*100;
imu_3.Magnetometer.NoiseDensity = 1.01*0.3/ sqrt(50);

% Random Number Generator
imu_3.RandomStream = 'mt19937ar with seed';
imu_3.Seed = 3;
```

GPS setup for Agent 1

```
fs_gps = 1; % Hz
refloc = [42.2825 -72.3430 53.0352]; % Quabbin resevoir in MA

gps = gpsSensor('UpdateRate', fs_gps);
gps.ReferenceLocation = refloc;
gps.DecayFactor = 0.5;
gps.HorizontalPositionAccuracy = 1.6;
gps.VerticalPositionAccuracy = 1.6;
gps.VelocityAccuracy = 0.1;

gps.RandomStream = 'mt19937ar with seed';
gps.Seed = 365;
```

Measurement Noises

```
Rmag = 0.09; % Magnetometer measurement noise
Rvel = 0.01; % GPS Velocity measurement noise
Rpos = 2.56; % GPS Position measurement noise
```

EKF setup for Agent 1

```
ekf_1 = insfilterMARG;
ekf_1.IMUSampleRate = fs_imu;
ekf_1.ReferenceLocation = refloc;

% Process noises
ekf_1.AccelerometerBiasNoise =  2e-4;
ekf_1.AccelerometerNoise = 2;
ekf_1.GyroscopeBiasNoise = 1e-16;
ekf_1.GyroscopeNoise =  1e-5;
ekf_1.MagnetometerBiasNoise = 1e-10;
ekf_1.GeomagneticVectorNoise = 1e-12;

% Initial error covariance
ekf_1.StateCovariance = 1e-9*ones(22);
```

EKF setup for Agent 2

```
ekf_2 = insfilterMARG;
ekf_2.IMUSampleRate = fs_imu;
ekf_2.ReferenceLocation = refloc;
```

```
% Process noises
ekf_2.AccelerometerBiasNoise =  2e-4;
ekf_2.AccelerometerNoise = 2;
ekf_2.GyroscopeBiasNoise = 1e-16;
ekf_2.GyroscopeNoise =  1e-5;
ekf_2.MagnetometerBiasNoise = 1e-10;
ekf_2.GeomagneticVectorNoise = 1e-12;

% Initial error covariance
ekf_2.StateCovariance = 1e-9*ones(22);
```

EKF setup for Agent 3

```
ekf_3 = insfilterMARG;
ekf_3.IMUSampleRate = fs_imu;
ekf_3.ReferenceLocation = refloc;

% Process noises
ekf_3.AccelerometerBiasNoise =  2e-4;
ekf_3.AccelerometerNoise = 2;
ekf_3.GyroscopeBiasNoise = 1e-16;
ekf_3.GyroscopeNoise =  1e-5;
ekf_3.MagnetometerBiasNoise = 1e-10;
ekf_3.GeomagneticVectorNoise = 1e-12;

% Initial error covariance
ekf_3.StateCovariance = 1e-9*ones(22);
```

Trajectory Flags

```
moving_traj_flag = true; % set true for LoggedQuadcopter data
```

Trajectory setup

```
if moving_traj_flag == true % Moving Traj
    % Load the "ground truth" UAV trajectory.
    load LoggedQuadcopter.mat trajData;

    % Agent 1
    trajOrient_1 = trajData.Orientation;
    trajVel_1 = trajData.Velocity;
    trajPos_1 = trajData.Position;
    trajAcc_1 = trajData.Acceleration;
    trajAngVel_1 = trajData.AngularVelocity;

    % Agent 2 offset 5 meters north Agent 1
    trajOrient_2 = trajData.Orientation;
    trajVel_2 = trajData.Velocity;
    trajPos_2 = trajData.Position + [5 0 0];
    trajAcc_2 = trajData.Acceleration;
    trajAngVel_2 = trajData.AngularVelocity;

    % Agent 3 offset 5 meters east Agent 1
    trajOrient_3 = trajData.Orientation;
    trajVel_3 = trajData.Velocity;
    trajPos_3 = trajData.Position + [0 5 0];
    trajAcc_3 = trajData.Acceleration;
    trajAngVel_3 = trajData.AngularVelocity;

else % Stationary Traj
    num_data = 20000;

    % Agent 1
    trajOrient_1 = quaternion(0,0,0,1)*ones(num_data,1);
    trajPos_1 = [2.5541 5.4406 0.75437].*ones(num_data,3);
    trajVel_1 = zeros(num_data,3);
    trajAcc_1 = zeros(num_data,3);
    trajAngVel_1 = zeros(num_data,3);

    % Agent 2 offset 5 meters north Agent 1
    trajOrient_2 = trajOrient_1;
    trajPos_2 = trajPos_1 + [5 0 0];
    trajVel_2 = trajVel_1;
    trajAcc_2 = trajAcc_1;
    trajAngVel_2 = trajAngVel_1;

    % Agent 3 offset 5 meters east Agent 1
    trajOrient_3 = trajOrient_1;
    trajPos_3 = trajPos_1 + [0 5 0];
    trajVel_3 = trajVel_1;
    trajAcc_3 = trajAcc_1;
    trajAngVel_3 = trajAngVel_1;
end
```

Initialize the states of the insfilterMARG for Agent 1

```
initstate_1 = zeros(22,1);
initstate_2 = zeros(22,1);
initstate_3 = zeros(22,1);
if moving_traj_flag == true
    initstate_1(1:4) = compact(meanrot(trajOrient_1(1:100)));
    initstate_1(5:7) = mean(trajPos_1(1:100,:), 1);
    initstate_1(8:10) = mean(trajVel_1(1:100,:), 1);

    initstate_2(1:4) = compact(meanrot(trajOrient_2(1:100)));
    initstate_2(5:7) = mean(trajPos_2(1:100,:), 1);
    initstate_2(8:10) = mean(trajVel_2(1:100,:), 1);

    initstate_3(1:4) = compact(meanrot(trajOrient_3(1:100)));
    initstate_3(5:7) = mean(trajPos_3(1:100,:), 1);
    initstate_3(8:10) = mean(trajVel_3(1:100,:), 1);
else
    initstate_1(1:4) = compact(trajOrient_1(1));
    initstate_1(5:7) = trajPos_1(1,:);
    initstate_1(8:10) = trajVel_1(1,:);

    initstate_2(1:4) = compact(trajOrient_2(1));
    initstate_2(5:7) = trajPos_2(1,:);
    initstate_2(8:10) = trajVel_2(1,:);

    initstate_3(1:4) = compact(trajOrient_3(1));
    initstate_3(5:7) = trajPos_3(1,:);
    initstate_3(8:10) = trajVel_3(1,:);
end
initstate_1(11:13) =  imu_1.Gyroscope.ConstantBias./fs_imu;
```

```
initstate_1(14:16) =  imu_1.Accelerometer.ConstantBias./fs_imu;
initstate_1(17:19) =  imu_1.MagneticField;
initstate_1(20:22) =  imu_1.Magnetometer.ConstantBias;

initstate_2(11:13) =  imu_2.Gyroscope.ConstantBias./fs_imu;
initstate_2(14:16) =  imu_2.Accelerometer.ConstantBias./fs_imu;
initstate_2(17:19) =  imu_2.MagneticField;
initstate_2(20:22) =  imu_2.Magnetometer.ConstantBias;

initstate_3(11:13) =  imu_3.Gyroscope.ConstantBias./fs_imu;
initstate_3(14:16) =  imu_3.Accelerometer.ConstantBias./fs_imu;
initstate_3(17:19) =  imu_3.MagneticField;
initstate_3(20:22) =  imu_3.Magnetometer.ConstantBias;

ekf_1.State = initstate_1;
ekf_2.State = initstate_2;
ekf_3.State = initstate_3;
```

Initialization of the Noise Characteristics

```
noiseMean = 0;
gps_variance = Rpos;
noiseVar = [0.01 0.01 0.25];
```